20 VIDEOLEZIONI DI BASIC
PER IMPARARE COL VIC 20

GRUPPO
EDITORIALE
JACKSON

- Joystick e paddle
- Informazione analogico-digitale
- Come controllare un Joystick - le interfacce
- LEN
- VAL
- RIGHT$ MID$ LEFT$
- Le operazioni sulle stringhe - slicing
- Videoesercizi
- Videogioco n° 9

9

COMMODORE
VIC 20

VIDEO BASIC VIC 20
Pubblicazione quattordicinale
edita dal Gruppo Editoriale Jackson
Direttore Responsabile:
Giampietro Zanga
Direttore e Coordinatore
Editoriale: Roberto Pancaldi
Autore: Softidea - Via Indipendenza 88 - Como
Redazione software:
Francesco Franceschini, Enrico Braglia,
Fabio Calanca
Segretaria di Redazione:
Marta Menegardo
Progetto grafico:
Studio Nuovaidea - Via Longhi 16 - Milano
Impaginazione:
Silvana Corbelli
Illustrazioni:
Cinzia Ferrari, Silvano Scolari
Fotografie:
Marcello Longhini
Distribuzione: SODIP
Via Zuretti, 12 - Milano
Fotocomposizione: Lineacomp S.r.l.
Via Rosellini, 12 - Milano
Stampa: Grafika '78
Via Trieste, 20 - Pioltello (MI)
Direzione e Redazione:
Via Rosellini, 12 - 20124 Milano
Tel. 02/6880951/5

Autorizzazione alla pubblicazione Tribunale di
Milano n° 422 del 22-9-1984
Spedizione in abbonamento postale Gruppo II/70
(autorizzazione della Direzione Provinciale delle
PPTT di Milano).
Prezzo del fascicolo L. 8.000
Abbonamento comprensivo di 5 raccoglitori L. 165.000
I versamenti vanno indirizzati a: Gruppo
Editoriale Jackson S.r.l. - Via Rosellini, 12
20124 Milano, mediante emissione di assegno
bancario o cartolina vaglia oppure
utilizzando il c.c.p. n° 11666203.
I numeri arretrati possono essere
richiesti direttamente all'editore
inviando L. 10.000 cdu. mediante assegno
bancario o vaglia postale o francobolli.
Non vengono effettuate spedizioni contrassegno.

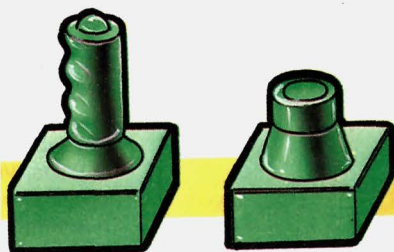
Gruppo Editoriale
Jackson

SOMMARIO

Introduzione

Chi non conosce il joystick, quella incandescente leva con pulsante onnipresente nei videogiochi da casa o da bar. Quante battaglie vinte o perse impugnandolo contro terribili nemici. Al di là del gioco, comunque il joystick (o le sorelle paddle) è una periferica di ingresso, con la sua logica (digitale) e i suoi principi di funzionamento, che è bene conoscere. Questo per quanto riguarda l'hardware.
Per la programmazione un altro prezioso tassello si aggiunge alle conoscenze già fatte: le funzioni di stringa.
Finora abbiamo limitato l'uso delle stringhe all'assegnazione, confronto e stampa. Vedremo in questa lezione come poter manipolare dei dati alfanumerici grazie a LEN, VAL, STR$, LEFT$, RIGHT$, MID$.

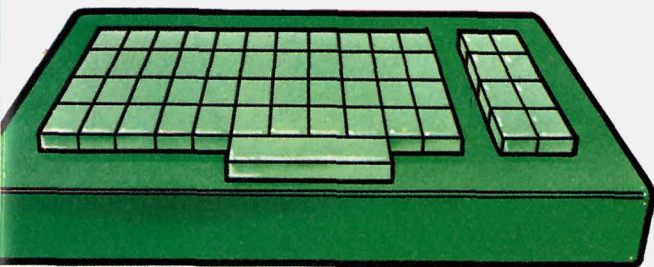
Paddle e Joystick

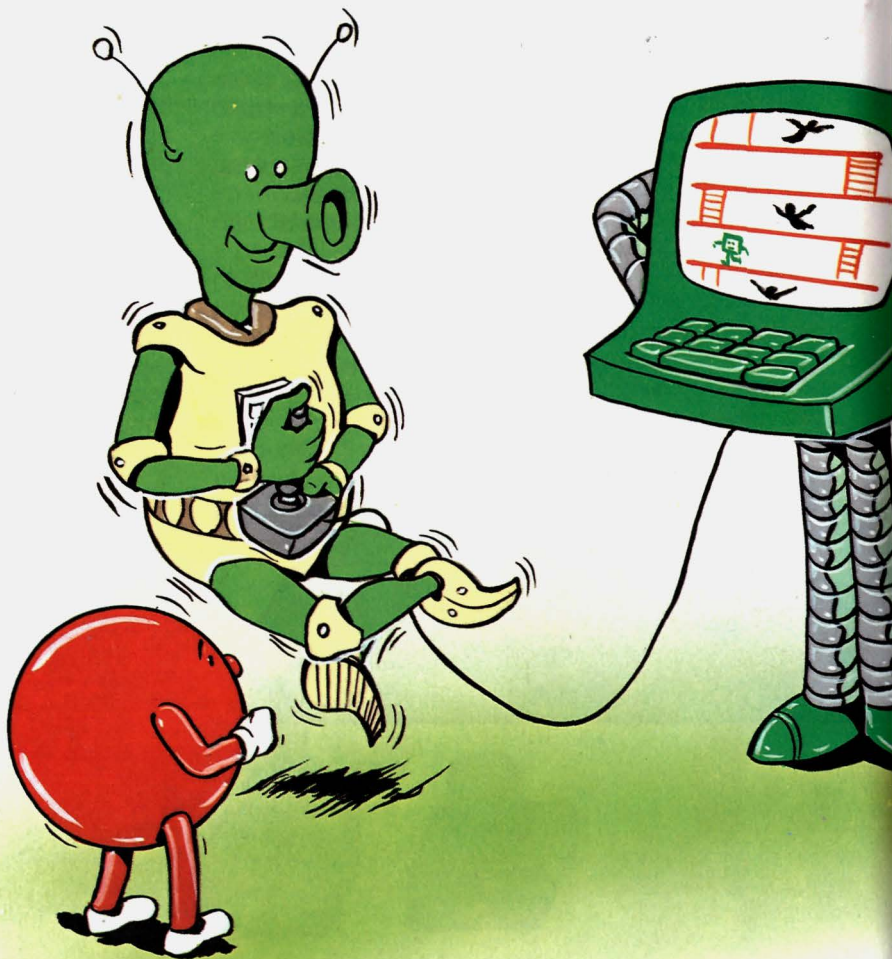
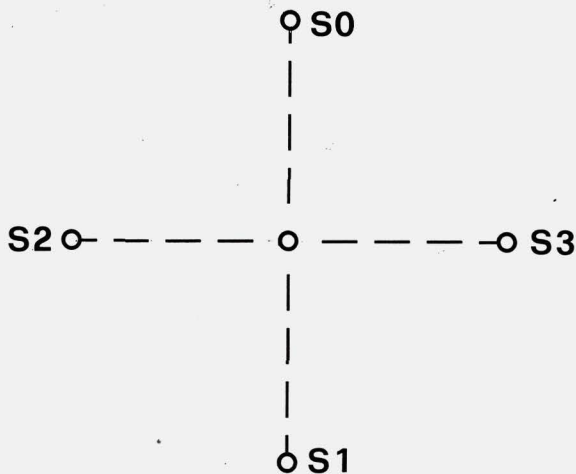
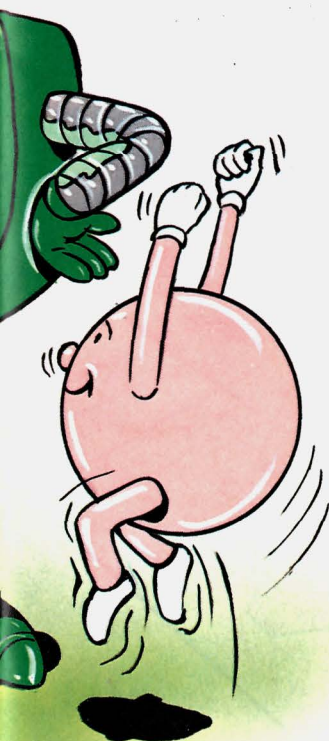
Abbiamo visto nelle scorse lezioni che il principale dispositivo di ingresso di dati ed informazioni per un calcolatore è costituito dalla tastiera, tant'è che praticamente non esistono computer - eccettuati quelli costruiti per particolari applicazioni - che non ne dispongono e che per l'input dei dati debbono ricorrere a mezzi differenti.
Principale non vuole però dire unico: sono infatti disponibili numerosi altri accessori e periferiche che in particolari circostanze possono rendersi molto più utili ed adeguati della tastiera, proprio come accade per l'uomo quando desidera qualcosa di più veloce o comodo della semplice (ma assolutamente indispensabile) penna per scrivere.
Oggi parleremo appunto di due di questi dispositivi: i joystick e le paddle.
Con ogni probabilità essi non ti sono completamente sconosciuti, dal momento che i loro nomi ricorrono molto spesso tra gli accaniti giocatori degli ormai diffusissimi ed onnipresenti videogiochi, essendo infatti utilizzati come principali armi di attacco e di difesa contro i vari invasori spaziali.
Anche chiunque sia entrato in un qualsiasi bar non può non conoscere un joystick, riconoscendo subito in tale nome l'incandescente levetta necessaria per evitare e sfuggire dai pericoli del "bombardamento

nemico".

Più facilmente non ti sarà invece molto noto il loro principio costruttivo e di funzionamento: l'obiettivo di questa lezione sarà appunto quello di spiegartene i segreti e le differenze.

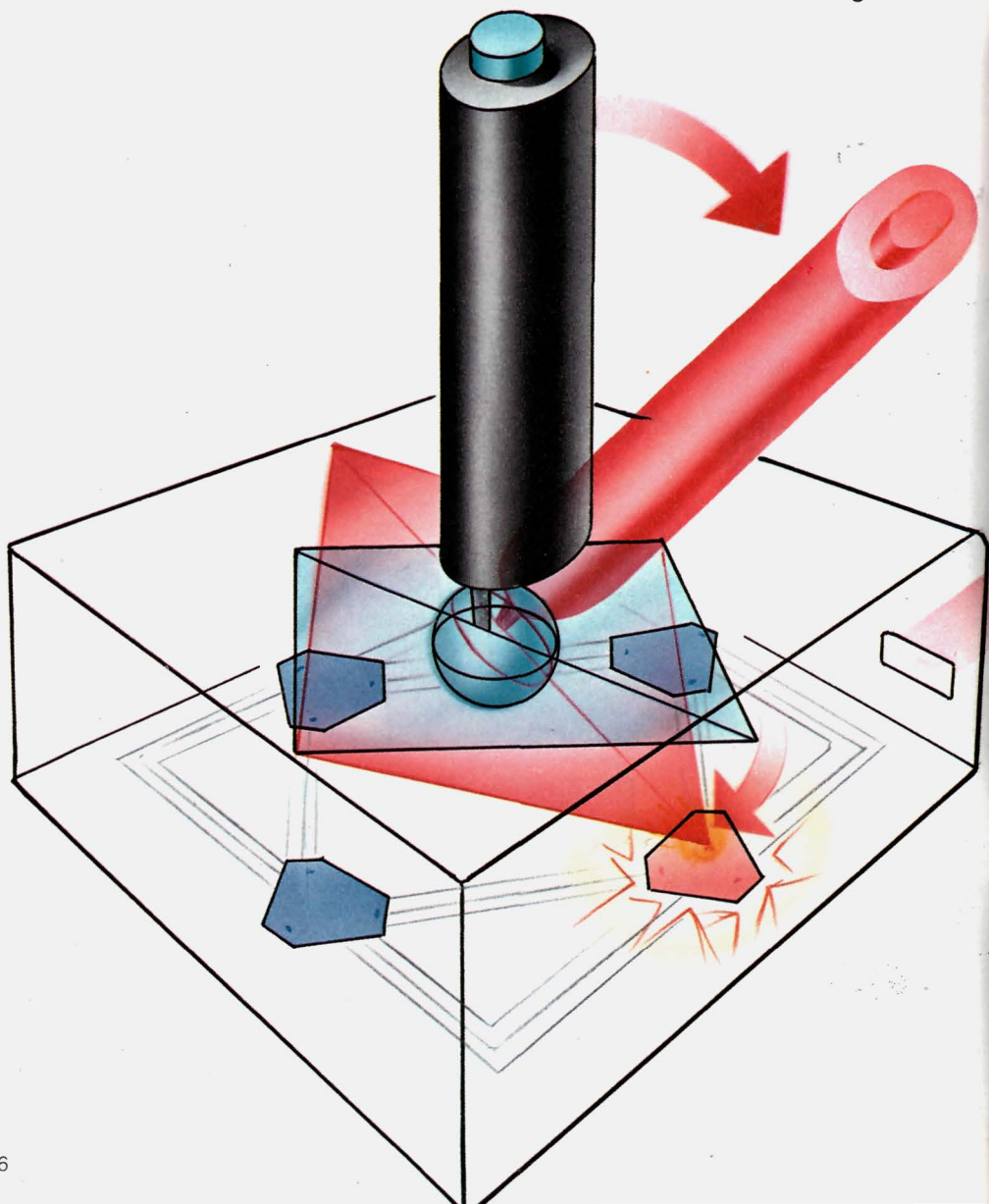
Cerchiamo innanzitutto di vedere quali sono stati i motivi che hanno determinato la nascita e, soprattutto negli ultimi tempi, la diffusione dei joystick e delle paddle.

In pratica, tutto è cominciato pochi anni fa, proprio con i fatidici videogiochi: un videogioco, infatti, altro non è che un particolare calcolatore programmato per eseguire solo ed esclusivamente un certo programma; cioè un calcolatore, come si usa dire, "dedicato". Il suo programma è appunto il gioco.

Non ci volle molto per rendersi conto che era necessario superare la difficoltà della tastiera: in primo luogo la maggior parte dei tasti sarebbe stata assolutamente superflua per l'uso al quale era destinato il calcolatore ed inoltre avrebbe reso molto più semplice, e soprattutto divertente, per gli utilizzatori poter avere l'impressione di "governare" direttamente, proprio come accade sulle astronavi vere, attraverso una specie di leva di comando ed un pulsante di fuoco. Attraverso questi accessori divenne pertanto possibile impartire ordini alla macchina, in modo rapido, facile e soprattutto senza conoscere nulla di programmazione.

Come controllare un joystick

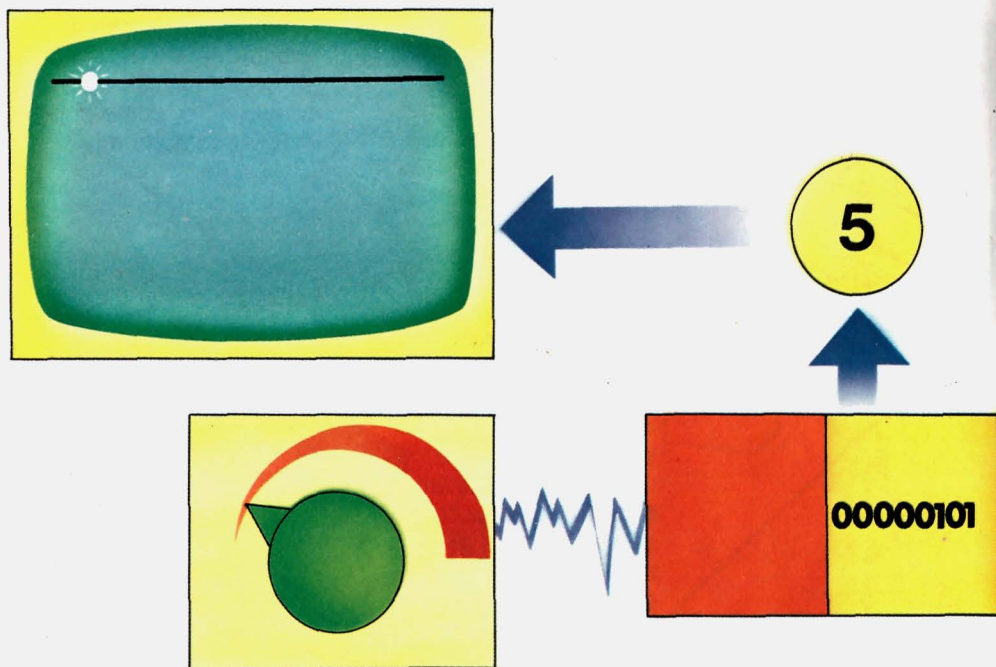
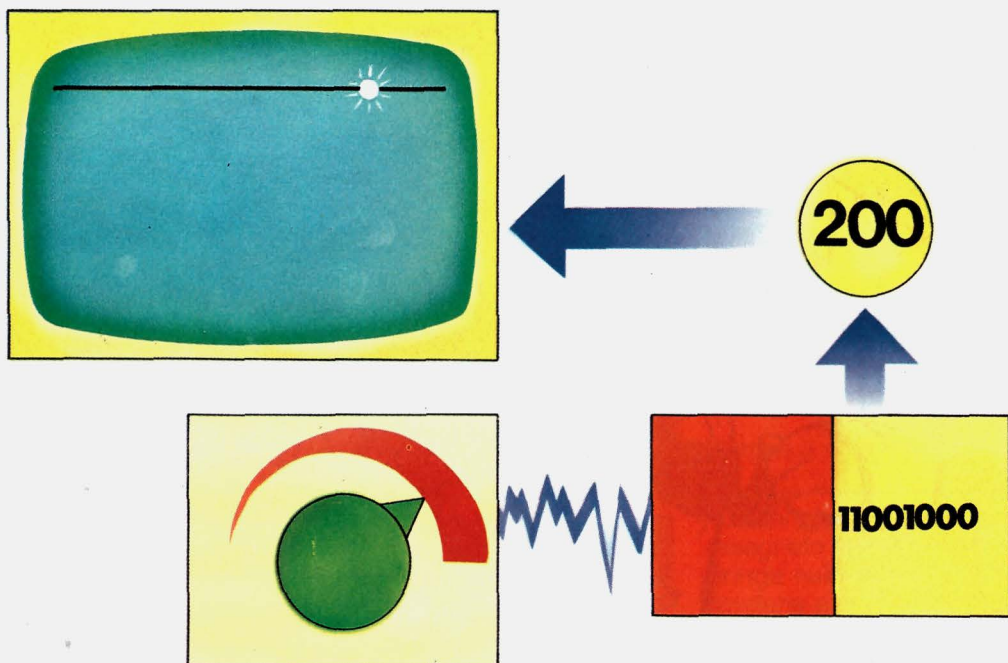
Per una persona non è molto naturale premere dei pulsanti per comandare dei movimenti: non si avrebbe la continua percezione di movimento che ricaviamo per esempio ruotando il volante di un'automobile se dovessimo invece utilizzare dei tasti o dei pulsanti.
Fu così che nacque l'idea dei joystick e, conseguentemente, delle paddle (che vedremo essere una specie di joystick piu raffinati).
Attualmente l'uso di questi dispositivi non è più prerogativa esclusiva

dei videogiochi: quasi tutti i *personal computer* ne permettono difatti la connessione, attraverso alcuni collegamenti appositamente realizzati sul calcolatore stesso. Un joystick è pertanto - alla stessa stregua della tastiera - un dispositivo di input; esso appare come una leva che con il suo movimento può permettere di spostare un oggetto sul video. Vediamo in quale modo. All'interno del joystick esistono quattro interruttori, che vengono *azionati in dipendenza* della direzione assunta dalla leva; in base al loro stato è perciò possibile identificare nove diverse posizioni della leva:

tutti aperti	leva verticale
S0 chiuso	nord
S1 chiuso	sud
S2 chiuso	ovest
S3 chiuso	est
S0, S1 chiusi	nord-ovest
S1, S2 chiusi	sud-ovest
S2, S3 chiusi	sud-est
S3, S0 chiusi	nord-est

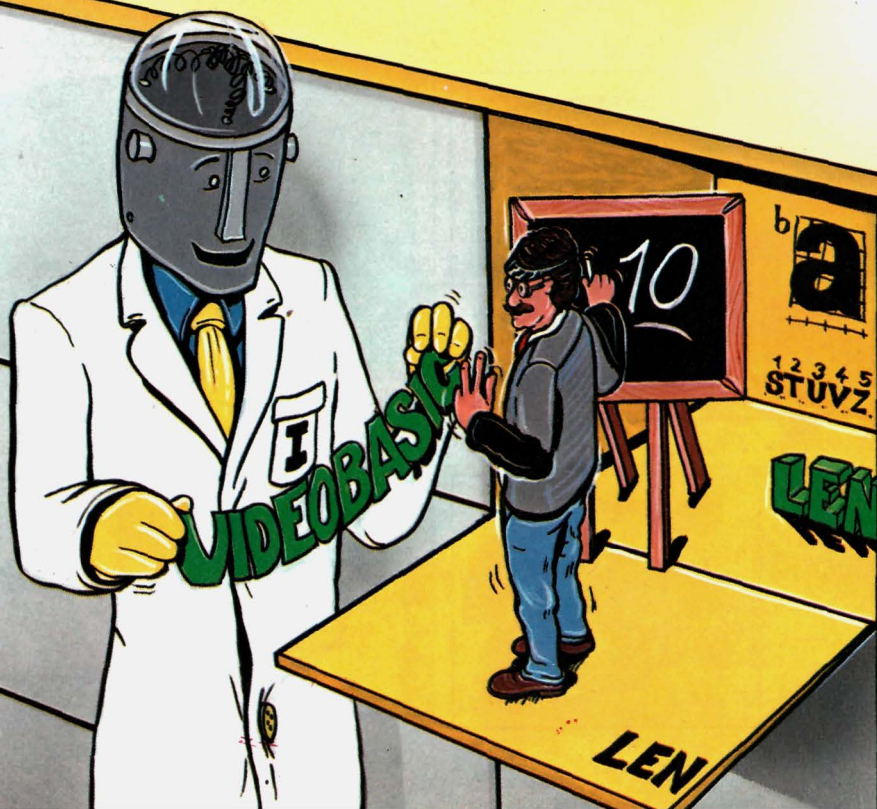
HARDWARE

A ciascuno di questi quattro interruttori viene associato, in una locazione della memoria all'interno del calcolatore, un bit: bit 1 per interruttore acceso o bit 0 per interruttore spento. Un quinto interruttore funge da

pulsante di fuoco o da segnalatore che la posizione desiderata è stata raggiunta.
Tramite programma basterà allora leggere nella locazione il valore di ciascuno di questi bit per risalire alla posizione della leva e quindi alla relativa azione da eseguire (spostamento del cursore, dell'astronave o dell'oggetto sul video).

Informazioni analogiche e digitali: interfaccia

Il joystick è pertanto un tipico dispositivo digitale: il dato contenuto nella celletta di memoria è infatti l'esatto corrispondente della combinazione di interruttori conseguente all'azionamento della leva. Lo stato di ciascun bit dipende cioè dallo stato del rispettivo interruttore, senza nessuna approssimazione od arrotondamento.
Talvolta risulta però più comodo poter disporre di un dispositivo più sensibile a piccole variazioni e spostamenti, proprio come accade con un volante di automobile. Al joystick propriamente detto è stato quindi affiancato il cosiddetto joystick a potenziometro (paddle).
Ai fini del funzionamento una paddle è molto simile ad un joystick: tuttavia, anziché esservi una leva, nella paddle compare una manopola, la cui regolazione risulta pertanto molto più fine e precisa. La cosa viene inoltre ottenuta in modo completamente diverso da come accade con il joystick: lo spostamento della manopola provoca infatti non più l'azionamento di interruttori e pulsanti, ma delle variazioni di tensione ai capi di due resistenze variabili (o potenziometri) inserite nella paddle stessa. Il risultato finale è pertanto un dato variabile con continuità o, come si usa dire, è costituito da

un'informazione analogica. Un'opportuna conversione di tale informazione da analogica in digitale fornisce quindi, sotto forma di numero a otto bit, l'azione specificata dallo spostamento della manopola, memorizzandola in una particolare locazione della memoria. Per eseguire questa conversione viene naturalmente utilizzata un'apposita interfaccia, posta tra unità centrale e paddle, che ha lo scopo di trasformare il segnale continuo fornito dalla paddle in un'informazione binaria. Attraverso un'operazione di lettura, in questa locazione sarà allora possibile, analogamente a quanto succede col joystick, risalire al movimento desiderato.

Naturalmente, anche una paddle ha un suo limite di sensibilità, dovuto proprio a tale operazione di conversione: esso risulta comunque estremamente al di sopra di quello del joystick, permettendo in totale 256 possibili combinazioni. Non è però assolutamente detto che una paddle sia meglio di un joystick; dipende infatti dall'uso e dalla sensibilità necessaria alla singola applicazione. Ciò che sicuramente è più semplice è la connessione del joystick al computer, poiché non richiede alcun dispositivo di conversione, come invece è il caso della paddle.

Gli operatori di stringa

Fino a questo momento il nostro utilizzo delle stringhe si è limitato soltanto all'assegnazione, al confronto ed alla stampa delle espressioni di questo tipo. In molti programmi per computer compaiono invece, in maniera estensiva, manipolazioni e trattamenti dei dati alfanumerici. In questa importante lezione cercheremo quindi di imparare le istruzioni e i comandi che consentono di effettuare simili operazioni.

LEN

La funzione LEN (abbreviazione dell'inglese LENght, lunghezza) è molto semplice da comprendere: consente infatti di trovare il numero di caratteri contenuti in una stringa.
Per esempio, volendo determinare il numero di caratteri che compongono la stringa "GIOVANNI", sarà sufficiente impartire:

```
PRINT LEN ("GIOVANNI")
```

e sullo schermo apparirà 8.
La stringa "GIOVANNI", della quale vogliamo contare i caratteri, è l'argomento di LEN e va quindi messa entro parentesi, subito dopo la parola LEN.
LEN restituisce quindi un risultato numerico, partendo da un argomento rigorosamente di tipo stringa, variabile o costante. Da notare che le virgolette non vengono considerate (e d'altra parte è anche giusto e logico) come costituenti la stringa.
È un comando molto potente e di utilizzo assai frequente: i possibili casi nei quali può interessare conoscere la lunghezza di una stringa sono infatti numerosissimi, tra i quali - per esempio - gli incolonnamenti, le impaginazioni o i confronti tra stringhe.

Esempi

```
PRINT LEN ("CONTO CORRENTE")
```

Stampa il numero 14.

```
LET A = LEN ("")
```

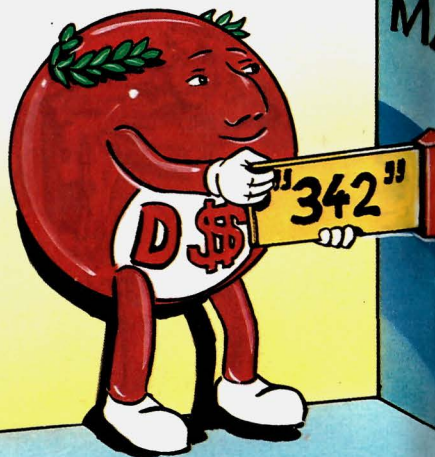
Alla variabile viene assegnato valore 0 (la stringa nulla non contiene alcun carattere).

```
LET A$ = "GIAN"
PRINT LEN (A$ + "CARLO")
```

```
LET A$ = "GIAN"
PRINT LEN (A$) + LEN ("CARLO")
```

Queste due istruzioni hanno lo stesso risultato, cioè 9. Nel primo caso viene prima eseguita la concatenazione delle due stringhe e quindi il calcolo della lunghezza, mentre nel secondo vengono separatamente calcolate le due lunghezze e quindi sommate.

Sintassi della funzione

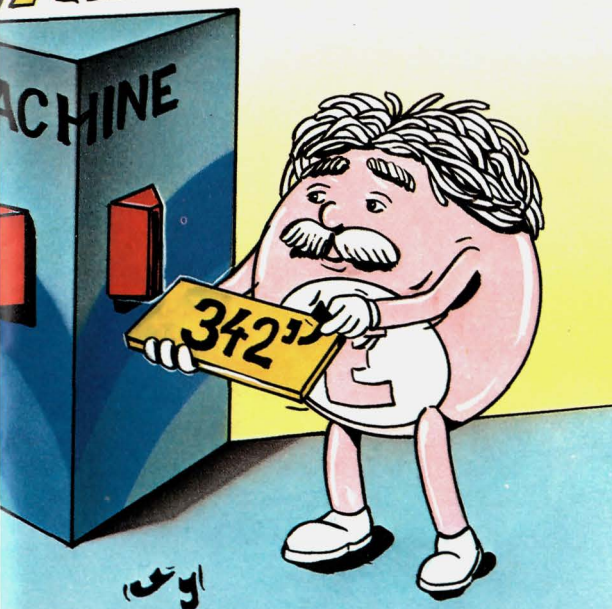
LEN (stringa)

VAL

Vi sono alcuni casi in cui risulta molto comodo poter convertire una stringa di numeri (come "3149") in un valore numerico, così da poterne far uso in espressioni numeriche. La funzione VAL (abbreviazione di VALue, valore) converte le stringhe in numeri. Così:

```
LET A = VAL ("3149")
```

assegna alla variabile numerica A il valore. 3149.
Non bisogna comunque fraintendere questa funzione: essa non esegue infatti alcuna magia. Una stringa come "TAPPO" non potrà pertanto mai assumere alcun valore numerico; quando l'interprete BASIC incontra un'istruzione come

```
PRINT VAL ("TAPPO")
```

convenzionalmente assegna allora al risultato valore 0, per quanto la cosa possa sembrare non aver molto senso.

Esempi

LET X = VAL ("8")	Assegna alla variabile X il valore *numerico* 8.
PRINT VAL ("54C3")	Stampa il numero 54, arrestando cioè la conversione al primo carattere non numerico.
LET P$ = "103" LET Q$ = "2" PRINT P$ + Q$	Stampa la somma (cioè la concatenazione) delle stringhe P$ e Q$. Quindi il risultato è la stringa "1032".
PRINT VAL (P$) + VAL (Q$)	Stampa la somma dei valori numerici 103 e 2, cioè 105.

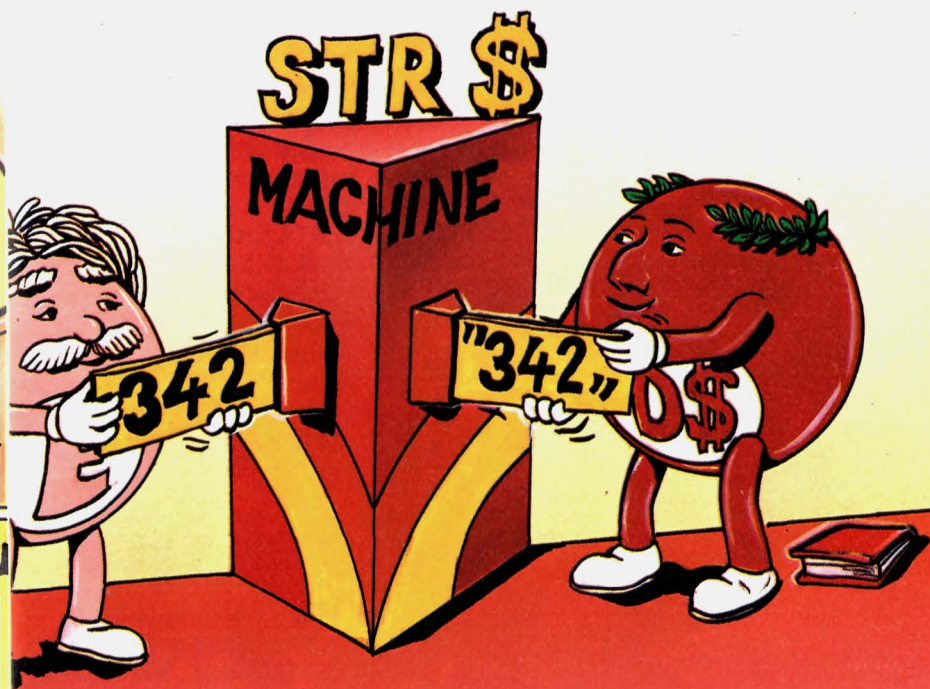
Sintassi della funzione

VAL (stringa)

STR$

STR$ (abbreviazione di STRing, stringa) è la funzione reciproca di VAL: essa consente infatti di convertire una quantità numerica in una stringa di caratteri. Quando desideriamo trasformare un numero (come 8) in una stringa, in modo che possa essere combinato con altre stringhe (per esempio "mila"), STR$ è quindi la funzione che fa al caso nostro. L'istruzione

```
PRINT STR$ (8) + " " + "MILA"
```

produrrà allora come risultato la stampa della stringa "8 MILA".

Esempi

LET A$ = STR$ (50)	Assegna ad A$ il valore alfanumerico "50".
LET S$= STR$ (3 * 7−2)	Risultato: S$ = "19".
PRINT VAL (STR$ (50))	È un'istruzione inutile, per quanto perfettamente lecita. Il valore numerico 50 viene infatti prima convertito in stringa (con STR$) e quindi ritrasformato in numero. Sullo schermo comparirà allora 50, inteso come numero, non come stringa.

Sintassi della funzione

STR$ (espressione)

LEFT$, RIGHT$

L'operazione di concatenamento permette di aggiungere caratteri ad una stringa (ricordandosi comunque di non superare il limite massimo di 255!), caratteri eseguendo cioè una sorta di somma tra stringhe.
In certi casi, anziché aggiungere, potrebbe essere necessario dover estrarre dei caratteri da una stringa: esistono allora nel linguaggio BASIC delle funzioni mediante le quali è possibile eseguire questa operazione di separazione in maniere diverse.
LEFT$ restituisce come risultato una stringa costituita dalla parte sinistra (left) della stringa utilizzata come argomento, per il numero di caratteri specificato.
Quindi:

```
PRINT LEFT$ ("GIANCARLO",4)
```

stampa i primi quattro caratteri della stringa "GIANCARLO", partendo dalla sinistra, cioè GIAN.
RIGHT$ ha le stesse regole di LEFT$, ma

esegue l'operazione dalla parte opposta: essa consente infatti di separare un certo numero di caratteri dalla stringa di partenza, cominciando questa volta dalla parte destra (right).
Allora

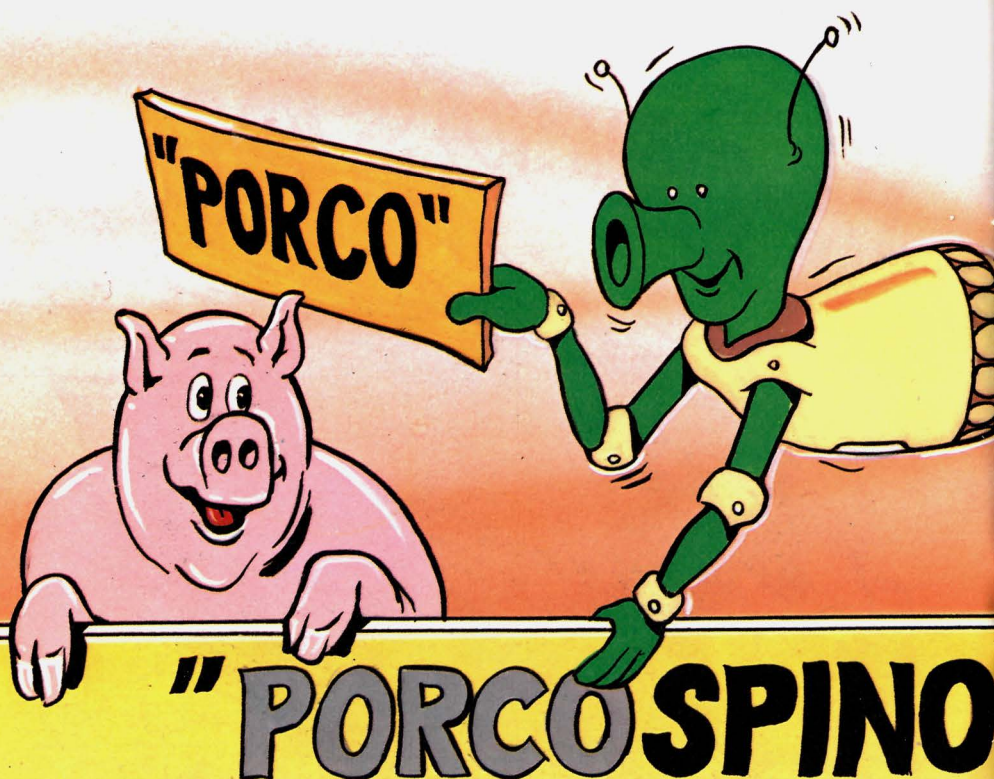
```
PRINT RIGHT$ ("GIANCARLO",5)
```

fornirà come risultato la visualizzazione sul video di "CARLO".
Nota come sia LEFT$ che RIGHT$ chiedano due argomenti: uno di tipo stringa ed uno di tipo numerico. Il primo specifica la stringa sulla quale si dovrà operare l'estrazione dei caratteri, il secondo indica invece il numero dei caratteri interessati dall'estrazione stessa.

Esempi

```
LET A$ = "gennaiofebbraiomarzo"
PRINT LEFT$ (A$, 7)
```

Stampa “gennaio”

```
LET A$ = "gennaiofebbraiomarzo"
PRINT RIGHT$ (A$, 5)
```

Stampa “marzo”

```
PRINT LEFT$ (("NASTRO"), LEN ("NASTRO") – 1)
```

Il numero di caratteri da prelevare partendo da sinistra sarà 5 (infatti LEN ("NASTRO") dà 6, che, diminuito di 1, risulta poi 5).
Sullo schermo apparirà allora "NASTR".

```
PRINT RIGHT$ (("NASTRO"), LEN ("NASTRO") – 1)
```

Tutto è uguale all'esempio appena visto, tranne che si parte da destra anziché da sinistra. Il risultato sarà allora la visualizzazione di "ASTRO".

```
PRINT RIGHT$ ("GIORNO", 7)
```

Poiché il valore 7 è maggiore della lunghezza totale della stringa, verrà stampata la stringa completa, cioè "GIORNO".

```
LET A$ = LEFT$ (A$, – 1)
```

Errore! Non è possibile estrarre – 1 caratteri da una stringa!

Il seguente, breve programma (altri ne troverai sulla parte della lezione dedicata alla programmazione) ti illustra una semplice applicazione dei concetti finora esposti:

```
10 INPUT "SCRIVI UNA PAROLA QUALSIASI"; P$
20 FOR I = 1 TO LEN (P$)
30 PRINT LEFT$ (P$, I); "  "; RIGHT$ (P$, I)
40 NEXT I
50 END
```

Sintassi delle funzioni

LEFT$ (stringa, numero caratteri)

RIGHT$ (stringa, numero caratteri)

MID$

MID$ permette l'estrazione di un certo numero di caratteri, partendo da un qualsiasi punto della stringa originaria. È sicuramente l'istruzione più potente del gruppo di comandi per la suddivisione delle stringhe (può infatti facilmente sostituire sia LEFT$ che RIGHT$).
MID$ si serve di tre argomenti: la stringa di origine, il numero che indica da quale carattere si deve iniziare l'estrazione ed infine il numero di caratteri da estrarre.

```
PRINT ("GRUPPO EDITORIALE JACKSON", 8, 10)
```

stampa quindi sullo schermo la stringa "EDITORIALE", prelevata dalla stringa di partenza, cominciando dalla ottava posizione e contando 10 caratteri.
L'utilizzo classico di questo comando lo si ha in quei programmi che richiedono in ingresso una data scritta nella forma GG/MM/AA (come 27/10/60). Con MID$ è allora molto facile scomporre la data nelle sue tre componenti GG, MM e AA.
Vediamo subito come fare:

```
10 INPUT "SCRIVI LA DATA (GG/MM/AA)"; D$
20 LET G$ = MID$ (D$, 1, 2)
30 LET M$ = MID$ (D$, 4, 2)
40 LET A$ = MID$ (D$, 7, 2)
```

Alla fine di questo programma G$, M$ e A$ conterranno rispettivamente il giorno, il mese e l'anno della data inizialmente assegnata a D$.

"PORCOSPINO"

Esempi

LET A$ = "gennaiofebbraiomarzo" PRINT MID$ (A$, 8, 8)	Stampa "febbraio" Nota che con MID$ si possono ottenere anche gli stessi risultati delle funzioni LEFT$ e RIGHT$:
LET A$ = "gennaiofebbraiomarzo" PRINT MID$ (A$, 1, 7)	Stampa "gennaio"
LET A$ = "gennaiofebbraiomarzo" PRINT MID$ (A$. 16, 5)	Stampa "marzo"
LET K$ = MID$ ("CUORE", 1, 8)	Assegna alla variabile K$ l'intera lunghezza di "CUORE" - partendo dalla prima posizione - visto che il numero di caratteri è inferiore agli 8 desiderati.
PRINT MID$ ("VIDEOBASIC", – 1, 3)	Dal momento che la posizione di un carattere in una stringa non può essere inferiore a 1, anziché un risultato, apparirà quindi sullo schermo un messaggio di errore.

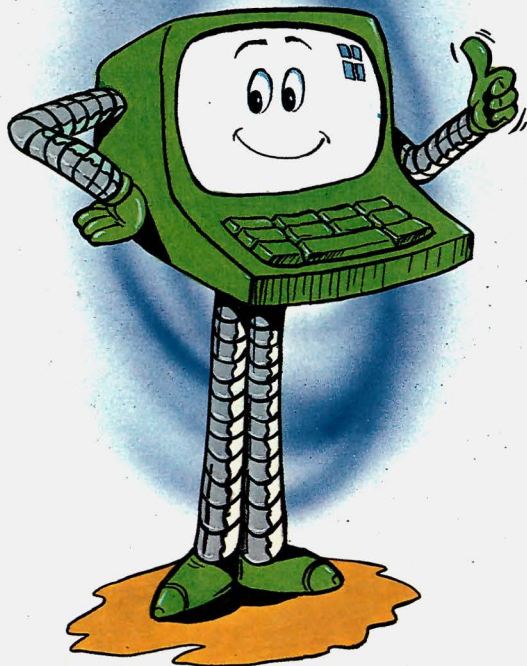
Sintassi della funzione

MID$ (stringa, posizione iniziale, numero caratteri)

Operazioni sulle stringhe

I programmi di oggi ti illustreranno alcune possibili applicazioni sulle funzioni per il trattamento delle stringhe.

Vediamo il primo esempio: stampare una stringa, introdotta dalla tastiera, solo se il primo carattere è diverso da "A" o da "B".

```
 10 PRINT "♥ INSERISCI LA STRINGA";
 20 INPUT N$
 30 REM A$ = DIVENTA LA PRIMA LETTERA DELLA STRINGA
 40 LET A$ = LEFT$ (N$, 1)
 50 REM CONTROLLA SE INIZIA PER A O B
 60 IF (A$ = "A") OR (A$ = "B") THEN GOTO 100
 70 REM LA STRINGA INIZIA CON UNA LETTERA DIVERSA
 80 PRINT "LA STRINGA È : "; N$
 90 END
100 PRINT "LA STRINGA NON È STAMPABILE PERCHÉ INIZIA PER "; A$
```

Particolari difficoltà non ne esistono; l'unica cosa da notare è la riga 60: è infatti necessario controllare che il primo carattere sia diverso da "A" o da "B".

Secondo esempio: scrivere una parola - come al solito introdotta dalla tastiera - in senso inverso, cioè opposto al normale (da destra verso sinistra).

```
10 PRINT "INSERISCI LA STRINGA";
20 INPUT A$
30 LET K = LEN (A$)
40 FOR I = K TO 1 STEP - 1
50 PRINT MID$ (A$, I, 1);
60 NEXT I : PRINT
70 INPUT "ANCORA"; R$
80 IF R$ = "S" THEN RUN
90 END
```

Anche in questo caso il procedimento è molto semplice: si preleva un carattere alla volta dalla parola (o dalla frase) da scrivere, partendo dalla destra e per tante volte quanti sono complessivamente i caratteri costituenti la parola stessa. Alla fine sullo schermo si troverà visualizzata la stringa originaria scritta al rovescio.
Se qualcosa non ti è chiaro, aggiungendo la riga

```
35 FOR J = 0 TO 500 : NEXT J
```

introducendo cioè un ciclo di ritardo, avrai la possibilità di osservare molto più agevolmente la formazione, carattere dopo carattere, della parola completa.
Un'altra possibile soluzione avrebbe potuto essere:

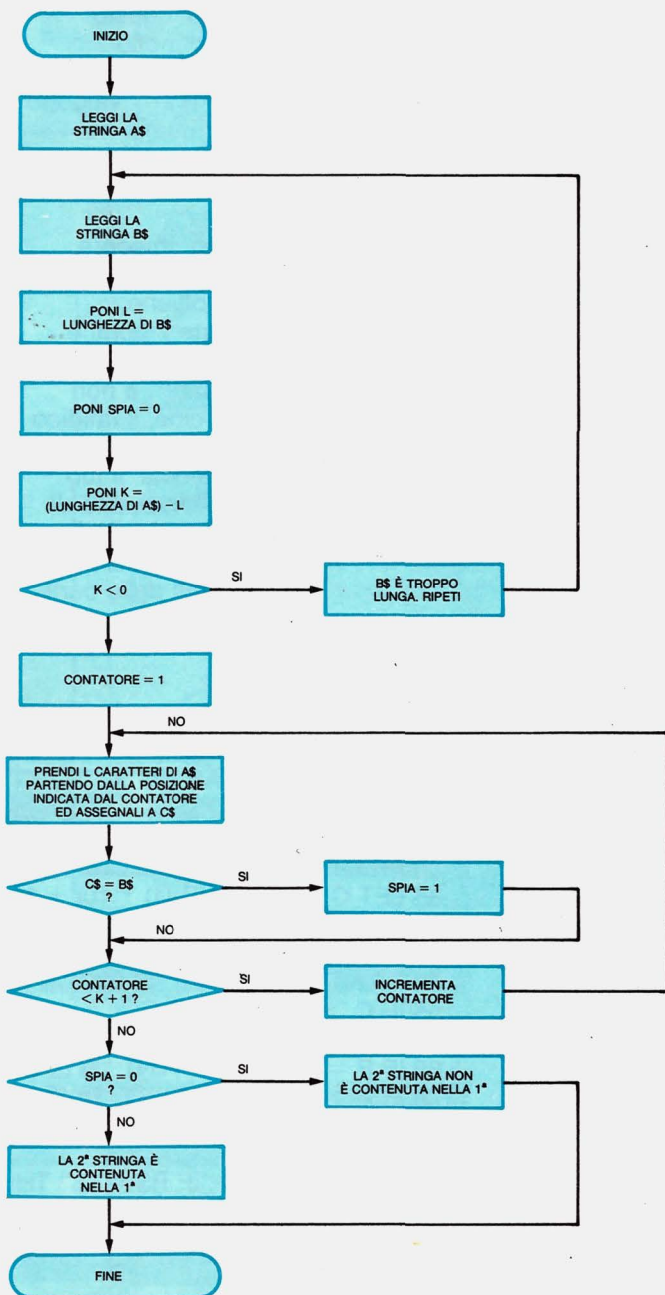
```
 10 PRINT "INSERISCI LA STRINGA";
 20 INPUT A$
 30 LET B$ = " "
 40 LET K = LEN (A$)
 50 FOR I = 1 TO K
 60 LET A$ = LEFT$ (A$, K - I + 1)
 70 LET C$ = RIGHT$ (A$, 1)
 80 LET B$ = B$ + C$
 90 NEXT I
100 PRINT B$
110 END
```

In questo caso i vari caratteri, anziché essere direttamente inviati sullo schermo, vanno ad aggiungersi, uno dopo l'altro, nella variabile B$. Terminato il ciclo (linee 30 - 70) B$ conterrà la parola invertita e la si potrà quindi stampare.

Come ultimo esempio vogliamo risolvere questo problema: cercare se una stringa è contenuta in un'altra stringa (esempio: la parola "carica" è contenuta in "scaricatori", mentre "mela" non ha nulla a che vedere con "pagina").
Vediamone innanzitutto una possibile risoluzione:

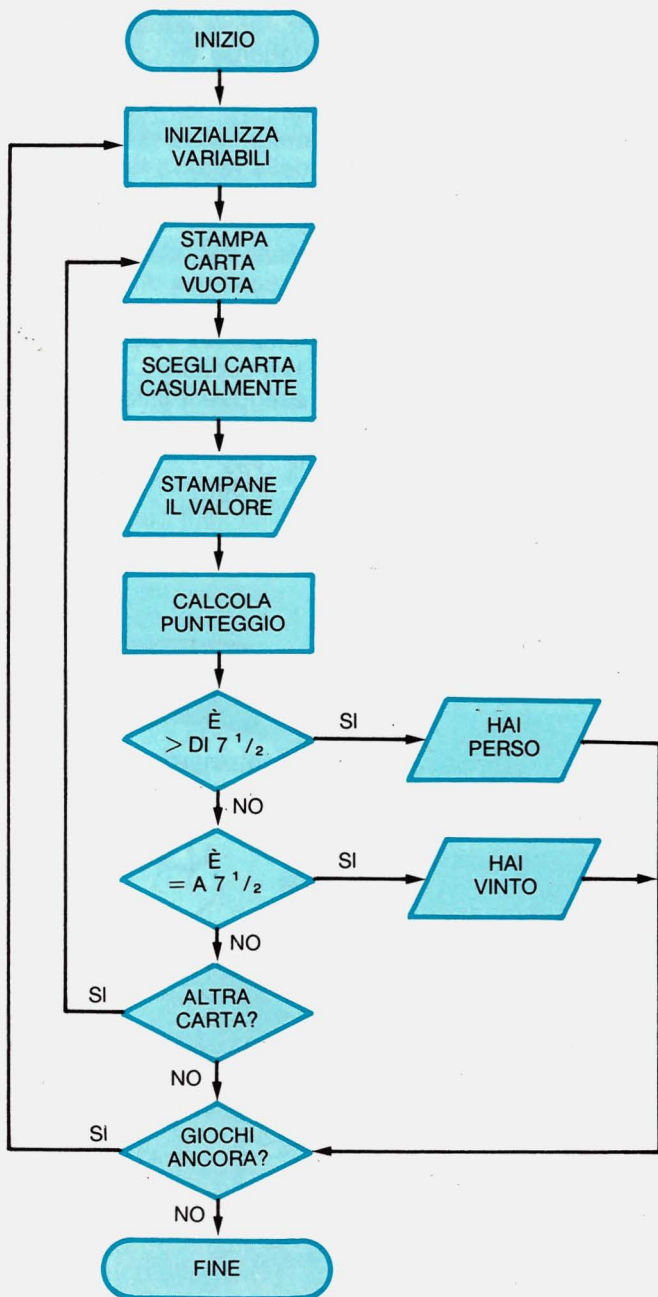
```
10 INPUT "■ INSERISCI LA PRIMA STRINGA"; A$
20 INPUT "INSERISCI LA SECONDA STRINGA"; B$
30 LETL = LEN (B$)
40 LET SPIA = 0
50 LET K = LEN (A$) - L
60 IF K < 0 THEN PRINT "TROPPO LUNGA! RIPETI"
   : GOTO 20
70 FOR I = 1 TO K + 1
80 LET C$ = MID$ (A$, I, L)
90 IF C$ = B$ THEN LET SPIA = 1
100 NEXT I
110 IF SPIA = 0 THEN PRINT "■ LA SECONDA STRINGA
   NON È CONTENUTA NELLA PRIMA" : END
120 PRINT "■" B$ "È CONTENUTA IN"; A$
```

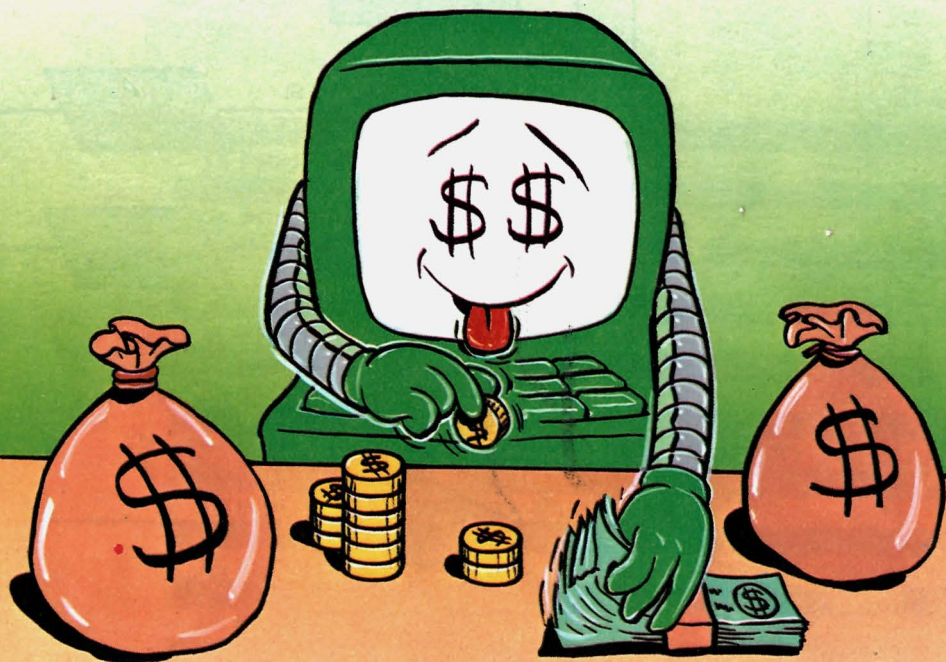
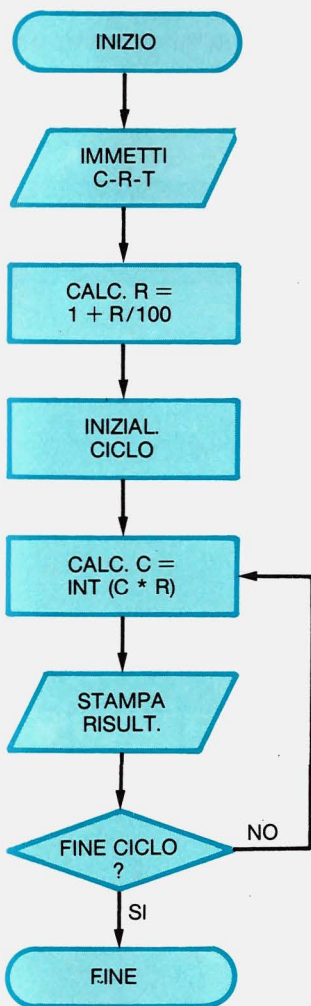
Ed ecco alcuni commenti esplicativi.
Righe 10 - 20: vengono richieste in ingresso la due stringhe, assegnandone i valori alle variabili A$ e B$.
Righe 30 - 40: la variabile numerica L assume il valore dato dal numero di caratteri

componenti la variabile B$. SPIA è invece una variabile che potrà assumere due soli valori, 0 e 1; varrà 0, se A$ non contiene B$, 1, nel caso contrario.
Righe 50 - 60: se la lunghezza di B$ risulta superiore a quella di A$, è chiaro che il confronto non può essere possibile. Occorre ripetere l'inserimento di B$.
Righe 70 - 100: questo è il cuore del programma. Utilizzando la funzione MID$ si confrontano man mano i caratteri di A$ con quelli di B$. Nel caso in cui l'esito sia positivo (cioè B$ sia contenuto in A$), FLAG assume valore 1.
Righe 110 - 120: costituiscono l'output del programma, sono cioè i messaggi di risposta al problema che ci eravamo proposti.

Settemezzo

Le regole di questo gioco sono molto semplici: le carte dall'asso (1) al 7 valgono tanti punti quanti ne rappresentano, mentre le "figure" soltanto $^1/_2$ punto.
Si possono chiedere quante carte si vuole e vince chi ottiene esattamente 7 punti e $^1/_2$. Attenzione, però, a non "sballare", a non superare, cioè, il fatidico punteggio.
Nel programma, il tuo VIC 20 ha il compito di gestire il gioco: si occupa, infatti, della distribuzione delle carte, del controllo scrupoloso del punteggio raggiunto e dell'invio dei messaggi appropriati.
Il cuore del problema sta nella simulazione dell'uscita delle carte. Questo compito è svolto dalle righe 10, 35, 40 del listato, mentre alla 15, 20, 25, 30, 45 e 50 è affidato quello di visualizzarle.
Tutto è basato sulla stringa "A234567JQK" assegnata ad A$ e sulle funzioni MID$ e RND per realizzare una corretta estrazione casuale.
Le righe dalla 55 alla 70 controllano il punteggio

```
10 LET A$ = "A234567JQR" : PRINT "■"
15 PRINT "■ ■" TAB (N) "┌────────┐"
20 FOR C = 1 TO 7
25 PRINT TAB (N) "|        |" : NEXT C
30 PRINT TAB (N) "└────────┘"
35 LET C = INT (RND (1) * 10) + 1
40 LET B$ = MID$ (A$, C, 1)
45 PRINT "■ ■" TAB (N + 1) B$
50 PRINT "■" TAB (N) "■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■" B$
55 IF C > 7 THEN LET C = .5
60 LET P = P + C
65 IF P = 7.5 THEN PRINT "■ ■ SETTEMMEZZO" : GOTO 90
70 IF P > 7.5 THEN PRINT "■ ■ HAI PERSO" : GOTO 90
75 INPUT "■ ■ UN'ALTRA CARTA"; R$
80 PRINT "■                    "
85 LET N = N + 3 : IF R$ = "S" THEN GOTO 15
90 INPUT "■ ■ UN'ALTRA PARTITA"; R$
95 IF R$ = "S" THEN RUN
```

e stampano i messaggi opportuni, mentre la 75, 80, 85, consentono di richiedere un'altra carta. L'ultima parte (righe 90 e 95) permette di giocare una nuova partita.

Capitali e interessi

Il prossimo listato riguarda un programma di tipo finanziario: il calcolo dell'interesse composto e la stampa della relativa tabella ordinata per anno.
L'algoritmo utilizzato è molto semplice: si tratta, in sostanza di:
A) Richiedere i dati necessari, vale a dire:
1) Il capitale (C) su cui eseguire il calcolo.
2) Il tasso di interesse (R) (associa il nome della variabile al termine < RAGIONE >)
3) Il tempo (T).
B) Calcolare il montante (capitale + interesse) per ogni singolo anno, avendo cura di considerare come capitale per l'anno successivo il montante del precedente.
C) Stampare il prospetto dei vari anni, avendo cura che gli importi risultino bene incolonnati.

```
 10 PRINT "♥"
 20 INPUT "CAPITALE"; C
 30 INPUT "INTERESSE"; R
 40 INPUT "ANNI"; T
 50 LET R = 1 + R/100
 60 FOR A = 1 TO T
 70 LET C = INT (C * R)
 80 PRINT A; TAB (21 – LEN (STR$ (C))); C
 90 NEXT A
100 END
```

Per quest'ultima esigenza, sarà molto comodo utilizzare le funzioni stringa < STR $ > e < LEN > unitamente a < TAB >.
Poiché nei tuoi futuri programmi incontrerai molto spesso la necessità di un incolonnamento rigoroso dei dati, è bene che familiarizzi ora con questo tipo di istruzioni.

Commento al listato

20-40 INPUT dei valori di CAPITALE, TASSO (R) e TEMPO (N. ANNI)
50 Trasformazione del tasso di interesse da percentuale a decimale
60 Impostazione del ciclo in funzione degli anni
70 Calcolo del montante annuo
80 Stampa della tabella con N. anno e valore montante.
Nota in particolare l'istruzione TAB (21 – LEN (STR$ (C))) che ha il compito di posizionare correttamente il dato da stampare.
90 Chiusura del ciclo.

VIDEOESERCIZI

Annota nello spazio apposito il risultato da te previsto per ciascun esercizio proposto e poi verificalo con la soluzione del tuo computer. Se avrai commesso anche un solo errore, ripassa la lezione.

```
10 LET A$ = "12345"
20 LET B$ = "67"
30 LET C$ = "890"
40 PRINT LEN (B$)
50 PRINT LEN (A$ + B$ + C$)
60 PRINT LEN (C$) – LEN (B$)
```

Per evidenziare

```
10 T$ = "TITOLO"
20 FOR C = 1 TO LEN (T$)
30 PRINT MID$ (T$, C, 1); " ";
40 NEXT C
```

Prima di far girare il programma scrivine l'output

Indovina dove stampa le stringhe

```
10 PRINT "♥"
20 INPUT "STRINGA ="; S$
30 PRINT "♥QQQQQQQQQQ"
40 PRINT TAB ((20 – LEN (S$))/2); S$
50 PRINT "QQQ ANCORA?"
60 GET R$ : IF R$ = " " THEN GOTO 60
70 IF R$ = "S" THEN RUN
80 END
```

In quale riga?

Dove rispetto alle colonne?